O programa é baseado na implementação de um servidor TCP que permite listar arquivos, realizar downloads, calcular hashes e continuar downloads a partir desse cálculo.

O código é baseado em um menu de 6 opções que aparecerá para o cliente. Dessa forma, ele poderá escolher entre as opções disponíveis qual funcionalidade ele deseja executar. As opções são:

1. listar arquivos
2. realizar downloads
3. realizar downloads de múltiplos arquivos através da máscara
4. calcular hashes
5. continuar downloads a partir do hash.
6. sair

Cada uma das opções escolhidas enviará um comando ao servidor (Ex: list, sget, etc), o qual interpretará e enviará as respostas ao cliente.

Ao longo de todo o código são criados try e except para tratar as exceções do código.

O código roda a partir de um loop e ao final de cada execução ele pergunta se deseja realizar uma nova operação.

Item A:Solicitar a listagem dos arquivos.

- O código verifica se o comando recebido pelo servidor corresponde à: "list". Caso corresponda, o servidor interpreta que o cliente deseja uma listagem de arquivos.
- É usado um print para mostrar uma mensagem indicando que o usuário solicitou os arquivos.
- É usado um try com um laço de repetição que: para cada arquivo na lista de arquivos presentes no servidor, verifica se o item é um arquivo comum usando os.path.isfile(caminho), obtém o tamanho do arquivo em bytes com os.path.getsize(caminho) e adiciona à lista. (se a lista estiver vazia, ele retorna que não encontrou nenhum arquivo)
- Por fim, usa o send para enviar a resposta ao cliente.

Item B: Download de arquivo

- O comando "sget" é enviado para o servidor utilizando o socket sock e codificado em UTF-8.
- O cliente recebe uma resposta de 2 bytes do servidor, que indica se o arquivo está disponível: if fileIsOk == 0, o arquivo existe no servidor e o processo de download é feito.
- Se o arquivo existir, o servidor enviará mais 8 bytes que correspondem ao tamanho do arquivo, sendo útil para o cliente saber que o servidor já terminou de enviar o arquivo.
- É usado: if os.path.exists(caminho_arquivo) para verificar se o arquivo já existe na lista de arquivos do cliente. Se sim, irá perguntar se deseja sobrescrever.
- São usadas variáveis: blocos_enviados e blocos_recebidos para monitorar o progresso em tempo real.

Item C: Download de múltiplos arquivos a partir de uma máscara

- O cliente fornece ao servidor a máscara que deseja. Ex: *.jpg
- O servidor utiliza o glob para encontrar arquivos no DIRBASE que correspondam à máscara fornecida.
- **No servidor:** para cada arquivo na lista, o servidor aguarda um byte de confirmação do cliente (b'\x01'), indicando que o cliente deseja baixar aquele arquivo, envia o tamanho do arquivo em 8 bytes e o conteúdo do arquivo.

- **No cliente:** para cada arquivo na lista, verifica se o arquivo já existe localmente, se existir, pergunta ao usuário se ele deseja sobrescrevê-lo, caso o usuário opte por não sobrescrever, o arquivo é ignorado. Envia um byte de confirmação (`b'\x01') ao servidor e recebe o tamanho do arquivo em 8 bytes.
- São usadas variáveis: blocos_enviados e blocos_recebidos para monitorar o progresso em tempo real.

Item D: Cálculo do hash de um arquivo

- O cliente envia o nome do arquivo e o número de bytes que serão utilizados no cálculo do hash. Se o arquivo não existir, uma mensagem de erro é enviada ao cliente.
- O servidor verifica a existência do arquivo, se o número de bytes é maior que 0 e se não excede o tamanho do arquivo. Caso o valor seja inválido, uma mensagem de erro é enviada ao cliente.
- Em seguida, o cálculo do hash é realizado utilizando o número de bytes informado pelo cliente. Se ocorrer algum erro durante o cálculo, uma mensagem de erro será gerada para o cliente.
- Após o cálculo do hash, o servidor envia o resultado ao cliente, que o recebe e exibe.

Item E: Continuação do download de um arquivo a partir do cálculo do hash

- O cliente envia a requisição "cget" ao servidor para indicar que deseja realizar um download continuado. O nome do arquivo é solicitado ao usuário e concatenado ao diretório DIRBASE para criar o caminho completo do arquivo no cliente.
- O cliente calcula o tamanho do arquivo local e envia o nome do arquivo e o tamanho local ao servidor para que este saiba de onde deve continuar o envio.
- O cliente calcula o hash SHA-1 da porção local do arquivo e envia ao servidor.
- Se o servidor retornar "HASH OK", isso indica que os hash é correspondente e que dados já baixados são válidos, continuando o download.
- O servidor calcula o tamanho total do arquivo e envia esse valor (8 bytes) ao cliente para determinar o fim do download.
- O servidor recebe o nome do arquivo, o tamanho local (em bytes) e o hash do cliente, verifica se o arquivo solicitado existe e calcula o hash SHA-1.

Item F: Limitar-se aos arquivos na pasta files

- Usa o os.path.realpath(caminho) para determinar o caminho absoluto para os arquivos que o servidor vai acessar. Assim, vai retornar o caminho absoluto da pasta base permitida, que é definida na variável DIRBASE.